1. var a =”12”;

Number(a) 强制转换成数字

若Var b=”12px” ，此时使用Number（）强制转换时NaN，应该使用parseInt（b）来转换，

ParseInt（）可以将字符串开头的几个数字转换为int，但如果开头不是数字结果是NaN。

2var c=["23","a","2","a"];

alert(typeof c); 此时显示 object

对数组对象而言，显示的结果就是object 而不是Array。

alert(c instanceof Array); true 判断c 是不是array的实例

3.js中非0就是true。

alert(0); --0

alert(!0); --true

alert(!!0); --false

但是特殊的 NaN 是false。

4.当一个变量没有定义值的时候他是undefined

alert(!!undefined); --false undefined类型是false

在js中除了NaN undefined 和 0， 其它都是true

5.js定义对象 ---是基于原型拷贝的

//定义对象

function Person(name,age){

this.name = name; --定义了为name的属性

this.age = age; -定义了为age的属性

this.address =”济南”;

//如果没有用this生命，这个变量只是一个局部变量，不是类的属性。

var b = “”；

this.say=function(){

}

}

//创建对象

Var person1 = new Person（“张三”，12）；

此时alert（typeof person1）； ----object

person1 instanceof Person --true

var x =function（）{

alert（“x”）；

}

//此时x就是一个function函数

x（）；

--------------------------------------------------------------

function fn（）{}

//此时是将y这个变量指向函数fn，可以通过y（）来调用函数。

var y=fn；

//只是取得返回值

var z=fn（）；

也可以用person1[“name”] 调用属性。

在js中对于对象，可以用for in 遍历对象显示声明的属性

for( var a in person1){

alert(a); --得到属性

alert（person1[a]）； --得到属性值

}

6.

document.write(); --在文档中打印

String对象的slice（）提取字符串

Substring（2,3）；从2开始取3个字符。

array

Js中的array是list和stack的集合

Var a = new Array（）；

a.push(11);

a = [1,2,”4”];

alert(a.join(“--”)); --可以转成字符串并以--来连接

sort（）； 可以排序 以字符串排序

Number

toFlexd（） --把数字转换成字符串，可指定小数

7.

<div onclick="aa(this);">dianji</div> --this指代这个标签

function aa(obj){

alert(obj); -- 可以用obj.innerHTML 得到标签内部的值

}

在js中设置样式 要用：XX.style.样式 来设置

如更改css样式，原来css中有-的，如 font-size， 要用驼峰标识。

Obj.style.fontSize =”12px”;

with（XX）

with(document){

write("b"); --with里边的代码以document来作为根对象，即不用

} --每个都写document了

8.

在js中不存在类，所以可以直接用object创建对象。

Var person = new Object（）； --然后可以自己添加属性和方法，但无法重复利用

Json方式： json就是JavaScript简单对象标签

通过逗号间隔

var person = {

name:"张三",

age:"23",

say:function(){

alert(this.name+this.age);

}

}

浏览器对象：

window：指窗口本身

history

Location

window.location.href=”a.html”; 跳转

window.open(URL,name,features,replace); --打开新窗口 参数都是可选的，features代表窗口的特征，name一样的话代表打开的都是在同一个窗口中显示

window.opener; ---获取父类窗口

9.

Dom

.innerHTML --节点的内容

.nodeName --节点名称

.nodeType --节点类型 （元素 1；属性 2 ；文本 3；注释 8 ；文档 9）

.nodeValue --节点的值 只是针对文本节点时有用

.firstChild …

事件的处理函数中默认会有一个event参数，用来获取相应的事件信息 。IE中是window.evnet 、。 event.type 事件类型

一般if(event=event||window.event){} 来判断处理

10.函数

函数是一个特殊的对象，是一个Function类的实例。在内存中是键值对的形式存储。

函数虽然是对象，但和对象有区别，对象是通过引用的指向完成赋值，函数通过对象的拷贝来赋值。即如果，fn1（）{} ； var fn2 = fn1; 此时fn1变了，fn2是不会随着fn1变化的。

函数的参数和调用无关，如果函数有一个参数，而调用时传了两个，仅仅只会匹配一个参数。即函数不存在重载。函数可以作为另一个函数的参数使用。

function fn1(arg){

var rel = function(num){

return arg + num;

}

return rel;

}

var f = fn1(11); --此时返回的是内部的函数。

alert(f(20)); ---打印31，

函数属性：

arguments ，通过此属性可以获取相应的参数值。是数组。就算调用时和函数的参数不对应也可以得到。

在arguments中有一个callee方法，arguments.callee(arg); 反向调用函数

例子：

function jiecheng(arg){

if(arg<=1) return 1;

else return arg \* jiecheng(arg-1);

--此处可以改为arg\* arguments.callee(arg-1); 降低了耦合

}

var f =jiecheng;

alert(f(4));

js中递归应该用这种方式。

this，

11.Ajax